LEVEL II

Stanford Department of Computer Science
Report No. STAN-CS-80-825

November 1980

REFINED ANALYSIS AND IMPROVEMENTS ON SOME FACTORING ALGORITHMS

by

C. P. Schnorr

Research sponsored by

Advanced Research Projects Agency (Defense)

AO3889
N3980C0132 and NSF/MCS 77 23738.
(NESC)

DEPARTMENT OF COMPUTER SCIENCE
Stanford University

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: STAN-CS-80-825 (14)	2. GOVT ACCESSION NO.: AD-A096348
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): REFINED ANALYSIS AND IMPROVEMENTS ON SOME FACTORING ALGORITHMS (6)	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): C. P. Schnorr (10)	8. CONTRACT OR GRANT NUMBER(s): N3980C0132, MCS7723738
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Stanford University, Department of Computer Science	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: AO 3889
11. CONTROLLING OFFICE NAME AND ADDRESS: (15) N00039-80-C-0132, ARPA Order-3889	12. REPORT DATE: Nov[illegible] 1980 (11)
	13. NUMBER OF PAGES: 31 (12) 32
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): NESC	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE. DISTRIBUTION UNLIMITED.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Factoring Algorithms
Computers
Mathematics
Monte Carlo

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

(U) By combining the principles of known factoring algorithms we obtain some improved algorithms which by heuristic arguments all have a time bound $O(\exp \sqrt{c \ln n \ln\ln n})$ for various constants $c \geq 3$. In particular, Miller's method of solving index equations and Shanks method of computing ambiguous quadratic forms with determinant $-n$ can be modified in this way. We show how to speed up the factorisation of n by using preprocessed lists of those numbers in $[-u, u]$

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

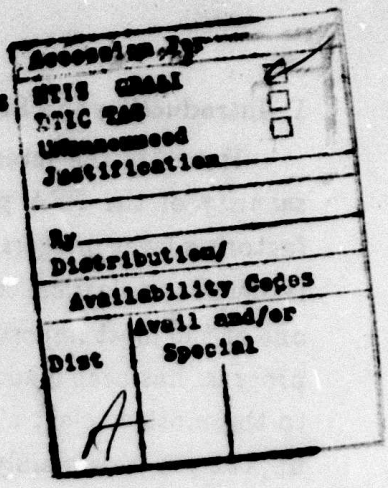
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

094120

and $[n-u, n+u]$, $0 << u << n$ which only have small prime factors. These lists can be uniformly used for the factorization of all numbers in $[n-u, n+u]$. Given these lists, factorization takes $O(\exp[2(\ln n)^{1/3}(\ln \ln n)^{2/3}])$ steps. We slightly improve Dixon's rigorous analysis of his Monte Carlo factoring algorithm. We prove that this algorithm with probability 1/2 detects a proper factor of every composite n within $O(\exp \sqrt{6 \ln n \ln \ln n})$ steps.

Refined Analysis and Improvements on Some Factoring Algorithms

C. P. Schnorr
Fachbereich Mathematik
Universität Frankfurt*



Abstract. By combining the principles of known factoring algorithms we obtain some improved algorithms which by heuristic arguments all have a time bound $O(\exp\sqrt{c \ln n \ln\ln n})$ for various constants $c \geq 3$. In particular, Miller's method of solving index equations and Shanks method of computing ambiguous quadratic forms with determinant $-n$ can be modified in this way. We show how to speed up the factorization of n by using preprocessed lists of those numbers in $[-u, u]$ and $[n-u, n+u]$, $0 << u << n$ which only have small prime factors. These lists can be uniformly used for the factorization of all numbers in $[n-u, n+u]$. Given these lists, factorization takes $O(\exp[2(\ln n)^{1/3}(\ln\ln n)^{2/3}])$ steps. We slightly improve Dixon's rigorous analysis of his Monte Carlo factoring algorithm. We prove that this algorithm with probability 1/2 detects a proper factor of every composite n within $O(\exp\sqrt{6 \ln n \ln\ln n})$ steps.

* This work was done in summer 1980 during a stay at the Stanford Computer Science Department. Preparation of this report was supported in part by National Science Foundation grant MCS-77-23738.

1. Introduction and Summary.

Recently the interest in factoring integers dramatically increased since the security of the RSA public key cryptosystem mainly relies on the difficulty of factoring large integers, see Rivest et al. (1978). The problem of factoring integers is one of the classical computational problems in mathematics. Gauss quoted it as one of the most important and most useful problems of arithmetics. Only modest progress has been made from the factoring methods known to Gauss and Legendre to the most efficient algorithms known today. In fact almost no new ideas came up, the progress mainly relies on more efficient programming and the use of faster computing machinery. Landmarks of this progress have been the factoring of the Fermat numbers $F_7 = 2^{2^7} + 1$ by Morrison and Brillhart (1975) and recently the factoring of $F_8 = 2^{2^8} + 1$ by Brent. The theoretical progress mainly concerns a better understanding and a more detailed analysis of the known methods. Also, with the evolution of the theory of computational complexity there evolved an increasing interest in asymptotical runtimes of algorithms. We will continue in this direction, too.

In order to factor n, or equivalently to solve $x^2 = -a \bmod n$, Gauss (Artikel 327) makes extensive use of the theory of quadratic forms. The usefulness of quadratic residues mod n which are small or only have small prime factors has been known long ago. Gauss (Artikel 328) gives a method to construct such residues w with $w = O(\sqrt{n})$ by means of quadratic forms. Legendre already used the continuous fraction expansion of $\sqrt{n}$. The more recent factoring algorithms of Morrison, Brillhart (1975), Shanks (1971, 1974), J. P. C. Miller (1975) are all refinements and variations of these old ideas. This will become clear by a comparative study of these algorithms, including proper modifications and improvements.

From the theoretical point of view Dixon (1978) achieved a major step. He proposed a probabilistic factoring algorithm and gave a rigorous proof that this algorithm for every composite number n with probability 1/2 detects a proper factor of n within $O(\exp(4\sqrt{\ln n \ln\ln n}))$ steps. Section 2 contains an outline of Dixon's analysis together with some improvements. In fact we decrease the constant 4 to $\sqrt{6}$. If in addition quadratic residues mod n are constructed via Legendre's continuous fraction method then, under reasonable assumptions, we obtain the time bound $O(\exp\sqrt{3\ln n \ln\ln n})$ for a tuned up version of the

Morrison–Brillhart algorithm.

In Section 3 we analyze J. P. C. Miller's method of using the solutions of index equations. We point out that this is not an independent method but rather a modification of solving $x^2 = y^2 \bmod n$ by combining congruences $\bmod n$. Under reasonable assumptions we obtain a time bound $O(\exp\sqrt{4.5 \ln n \ln\ln n})$. However this algorithm might be the most efficient one, if one likes to factor many numbers in a small region. The reason is that this algorithm uses lists of those numbers in $[-u, u]$ and $[n-u, n+u]$ which only have small prime factors. These lists can be uniformly used for the factorization of all numbers in $[n-u, n+u]$.

In Section 4 we modify Shanks (1971) method of factoring n via the construction of ambiguous quadratic forms with determinant $-n$. Our modification relates this algorithm to the previous ones and in particular to the Morrison–Brillhart algorithm. Under reasonable assumptions we obtain the time bound $O(\exp\sqrt{3 \ln n \ln\ln n})$.

This latter algorithm, the Morrison–Brillhart algorithm and the Schroeppel algorithm (see Monier (1980)) are the asymptotically fastest known factoring algorithms. A rough analysis slightly favors Schroeppel's algorithm since under reasonable assumptions we obtain a time bound $O(exp(1.5\sqrt{\ln n \ln\ln n}))$. There is however an additional speed up for the other two algorithms, due to the fact that about half of the primes cannot occur as factors of the residues occuring in the algorithm. This effect is difficult to analyze but might well dominate for all reasonably sized n over the small difference $\sqrt{3} - 1.5$ in the exponent.

We should at least mention the important algorithms of Pollard (1975) and of Schroeppel (see Monier, 1980) which are not included in this comparative study. For more complete surveys on factoring algorithms we recommend Guy (1975), *The Art of Computer Programming*, Vol. 2 by D. Knuth (in particular the 1980 edition), and the thesis of L. Monier (1980).

2. A Refined Analysis of Dixon's Probabilistic Factoring Algorithm.

So far the asymptotically fastest run time of a factoring algorithm has been proved by Dixon (1978). Given a composite number n, this algorithm finds a proper factor of n with probability 1/2 within $O(\exp(4\sqrt{\ln n \ln\ln n}))$ steps. ln denotes the "logarithmus naturalis" with the Eulerian number e as base and exp is the inverse function to ln. Dixon mainly applies the method of "combining congruences" to generate solutions of $x^2 = y^2 \bmod n$. In Sections 3 and 4 we will see that this technique can well be combined with factoring algorithms proposed by J. P. C. Miller (1975) and D. Shanks (1971). We give an outline of Dixon's algorithm with an improved analysis. We decrease the constant 4 in Dixon's bound to $\sqrt{6}$. The improved theoretical time bound results from a tighter lower bound on the number of quadratic residues $\bmod n$ which can be completely factored over small primes (Lemma 1) and a specific method for detecting small prime factors. Here we do not focus on designing the most practical algorithm but we like to prove a rigorous asymptotical time bound as small as possible.

Dixon's Algorithm.

begin input n

stage 1 $v = \lfloor n^{1/2r} \rfloor$
comment the optimal choice of $r \in \mathcal{N}$ will be made below.
Form the list P of all primes $\leq v$. $P = \{p_1, \ldots, p_{\pi(v)}\}$.
if $\exists p_i \in P : p_i \mid n$ then print p_i stop
$B := \emptyset$

stage 2 Choose $z \in [1, n]$ at random and independently from previous choices of z.
$w := z^2 \bmod n$ with $0 \leq w < n$

stage 3 Compute $\underline{a} = (a_i \in \mathcal{N} \mid 1 \leq i \leq \pi(v))$ and w^* with $w = w^* \prod_{i \leq \pi(v)} p_i^{a_i}$ and $\forall p \in P$: p does not divide w^*.

test 1 if $[w^* \neq 1$ or $\underline{a} = \underline{0} \bmod 2]$ then goto stage 2
$B := B \cup \{\underline{a}\}$, $z_{\underline{a}} := z$
Try to find a nontrivial solution of

$$\sum_{\underline{a} \in B} f_{\underline{a}} \underline{a} = \underline{0} \bmod 2 \quad f_{\underline{a}} \in \{0, 1\}. \tag{1}$$

test 2 **if** there is no nontrivial solution **then goto** stage 2
$x := \prod_{f_{\underline{a}}=1} z_{\underline{a}}$, $y := \prod_{i \le \pi(v)} p_i^{(\sum_{\underline{a} \in B} f_{\underline{a}} a_i)/2}$
comment [The construction implies $x^2 = y^2 \bmod n$; in case $x \not\equiv \pm y \bmod n$, $\gcd(x \pm y, n)$ are proper factors of n.]

test 3 **if** $x \not\equiv \pm y \bmod n$ **then** print $\gcd(x \pm y, n)$ **stop**
Choose the first $\underline{a} \in B$ such that $f_{\underline{a}} = 1$.
$B := B - \{\underline{a}\}$, **goto** stage 2

end

Obviously a proper factor of n has been found as soon as test 3 succeeds. In the following analysis of the algorithm we suppose that n is an odd number with prime factor decomposition.

$$n = \prod_{i=1}^{d} q_i^{\ell_i} \quad \ell_i \ge 1 \text{ and } d \ge 2.$$

Clearly the cases that n is even or a pure prime power can easily be handled in advance. The following facts are due to Dixon.

Fact 1. *prob*$(x = \pm y \bmod n$ *within test 3*$) = 2^{1-d}$ *and the corresponding events for distinct passes of test 3 are mutually independent.*

Proof. Consider the last chosen z and $w = z^2 \bmod n$. We prove that there are exactly 2^d distinct z_i, $i = 1, \ldots, 2^d$, such that $z_i^2 = w \bmod n$. Clearly Z_n^*, the multiplicative group mod n, is a direct sum

$$Z_n^* = \bigoplus_{i=1}^{d} Z_{q_i^{\ell_i}}^*.$$

For each i there are exactly two distinct solutions $t_i = u_i, v_i$ of $t_i^2 = w \bmod q_i^{\ell_i}$. Then by the Chinese remainder theorem the z_i correspond in one-one manner to the 2^d elements in $\{u_1, v_1\} \times \cdots \times \{u_d, v_d\}$. Now each of $z_1, \ldots, z_{2^d}$ is equally likely to be chosen for z. The values of $f_{\underline{a}}$ and y do not depend on the choice of $z \in \{z_1, \ldots, z_{2^d}\}$, only $x = \prod_{f_{\underline{a}}=1} z_{\underline{a}}$ depends on this choice. Observe that the value $f_{\underline{a}}$ corresponding to $z = z_{\underline{a}}$ must be 1, otherwise the algorithm would

pass test 3 without choosing this final z. Therefore the 2^d choices for z yield 2^d distinct values for x and exactly two of them imply $x = \pm y \bmod n$. This evaluates the probability that "$x = \pm y \bmod n$ during test 3" to 2^{1-d}. Since our analysis is completely based on the last chosen z, it is clear that the distinct events of "test 3 succeeding" are mutually independent. ∎

Let $T(n)$ be the total time of the algorithm and let $T_3(n)$ be the time till the first pass of test 3. We count arithmetical steps mod n as single steps. $T(n)$, $T_3(n)$ are random values depending on the random variables z of stage 2. Fact 1 immediately implies:

Fact 2. $E[T(n)] = (1 - 2^{1-d})^{-1} E[T_3(n)] \leq 2E[T_3(n)]$.

Here $E[X]$ denotes the expectation of the random value X. Let $T_1(n)$ ($T_2(n)$, resp.) be the time spent from *any* entering of stage 2 till the first pass of test 1 (test 2, resp.) without counting the steps used to solve the various linear systems of equations (1). Since a linear dependence of the $\underline{a}$ with $\underline{a} \in B$ must exist as soon as $\#B \geq \pi(v) + 1 = O(v/\ln v)$ it follows that there are almost $\pi(v) + 1$ passes of test 2 between two consecutive passes of test 3. Hence

Fact 3. $E[T_3(n)] \leq (\pi(v) + 1) E[T_2(n)] + O(\pi(v)^3)$.

Here $O(\pi(v)^3)$ bounds the steps to solve all the linear systems (1) occuring in the various passes of stage 3. Indeed this task amounts to solve *one* system of linear equations with $\pi(v) + 1$ unknowns. In order to analyze $E[T_2(n)]$ we define

$$Q := \{\text{set of quadratic residues mod } n\} \cap Z_n^*$$
$$T(n, v) := \{r \in [1, n]: \text{all prime factors of } r \text{ are} \leq v\}$$
$$M(n, v) := \{z \in [1, n]: z^2 \bmod n \in Q \cap T(n, v)\}.$$

Fact 4. $E[T_2(n)] \leq O(E[T_1(n)] n / \#M(n, v))$.

Proof. We clearly have $prob(w^* = 1) \geq \#M(n, v)/n$ and it can easily be seen that $prob(\underline{a} = \underline{0} \bmod 2)$ is negligibly small. Hence test 1 will almost be passed about $n/(\#M(n, v))$ times between two passes of test 2.

$T_1(n)$ depends on how the factorization of w over the prime base P is done. A crude way is as follows:

$w^* := w$
for all $p \in P$ **do**
[**while** $p \mid w^*$ **do** $w^* := w^*/p$]

This yields

Fact 5. $E[T_1(n)] \leq \pi(v) + \log n$.

Here $\log n$ bounds the number of multiple prime factors of n according to their multiplicity.

So far Facts 1–5 yield under the assumption $\log n \leq \pi(v)$:

$$E[T(n)] \leq O\left(\pi(v)^2\left[\frac{n}{\#M(n,v)} + \pi(v)\right]\right) \tag{2}$$

and it remains to prove a sharp lower bound on $\#M(n,v)$. This will be our main improvement over Dixon's analysis. Let κ: $Z_n^* \to \{\pm 1\}^d \approx \bigoplus_{i=1}^d Z_2$ be the quadratic character, defined as follows. For $(a_1, \ldots, a_d) \in \bigoplus_{i=1}^d Z_{q_i^{t_i}}^*$, let $\kappa(a_1, \ldots, a_d) = (e_1, \ldots, e_d)$ with $e_i = \left(\frac{a_i}{q_i^{t_i}}\right)$. By definition the Jacobi symbol $\left(\frac{b}{q}\right)$ is 1, (-1, resp.) if b is a quadratic residue (non-residue) mod q. It is well known that $\kappa : Z_n^* \to \bigoplus_{i=1}^d Z_2$ is a group homomorphism and $a \in Q$ iff $\kappa(a)$ is the group unit $(1, 1, \ldots, 1) \in \{\pm 1\}^d$.

Lemma 1. *$\#M(n,v) \geq \pi(v)^{2r}/(2r)!$ for all natural numbers r with $v^{2r} \leq n$ provided all prime factors of n are $> v$.*

Proof. Let $T_r(m,v) := \{w \in [1,m] \mid w = \prod_{p_i \leq v} p_i^{a_i} \wedge \sum_i a_i \leq r\}$. Since all prime factors of n are $> v$ we have $T_r(\sqrt{n}, v) \subset Z_n^*$. We partition $T_r(\sqrt{n}, v)$ into classes T_i, $i = 1, \ldots, 2^d$ according to the 2^d possible values of κ. Then

$$\bigcup_{i=1}^{d} T_i T_i \subset T_{2r}(n,v) \cap Q.$$

Therefore

$$\begin{aligned} \#M(n,v) &\geq 2^d \#(T_{2r}(n,v) \cap Q) \\ &\geq 2^d \sum_{i=1}^{2^d} \#T_i^2 \frac{r!^2}{(2r)!}. \end{aligned} \tag{3}$$

Here $(\#T_i)^2$ counts the number of ordered pairs $(w_1, w_2) \in T_i \times T_i$ and $(2r)!/(r!)^2$ bounds for each $w \in Q$ the number of distinct pairs $(w_1, w_2) \in \bigcup_i T_i \times T_i$ that yield the product $w_1 w_2 = w$. The Cauchy Schwarz inequality implies

$$\sum_{i=1}^{2^d} (\#T_i)^2 \geq 2^{-d} \Big(\sum_i \#T_i \Big)^2 = 2^{-d} \#T_r(\sqrt{n}, v)^2$$

$$\text{(use } \sum_i u_i^2 \cdot \sum_i v_i^2 \geq (\sum_i u_i v_i)^2 \text{ with } u_i = \#T_i, \quad v_i = 1).$$

Obviously we have $\#T_r(\sqrt{n}, v) = \binom{\pi(v)+r}{r} \geq \pi(v)^r / r!$, since $\binom{\pi(v)+r}{r}$ is the number of possibilities of choosing with repetitions r elements out of $\pi(v)$. Finally we obtain from (3), (4):

$$\#M(n, v) \geq \#T_r(\sqrt{n}, v)^2 \frac{r!^2}{(2r)!} \geq \frac{\pi(v)^{2r}}{r!^2} \frac{r!^2}{(2r)!} = \frac{\pi(v)^{2r}}{(2r)!}. \quad \blacksquare$$

Putting (2) and Lemma 1 together we obtain

$$E[T(n)] = O\Big(\pi(v)^2 \Big(\frac{n(2r)!}{\pi(v)^{2r}} + \pi(v) \Big) \Big)$$

provided $\log n \leq \pi(v)$ and $v^{2r} \leq n$. Using $v = n^{1/2r}$, the prime number theorem in the form $v/\ln v \leq \pi(v) \leq 2v/\ln v$ and Stirling's formula

$$(2r)! = O(\sqrt{2r}(2r)^{2r} e^{-2r})$$

we obtain

$$E[T(n)] = O\Big(\frac{(4r)^2 n^{1/r}}{(\ln n)^2} \Big[\sqrt{2r}\, e^{-2r} (\ln n)^{2r} + \frac{4r n^{1/2r}}{\ln n} \Big] \Big).$$

We choose $r \in \mathcal{N}$ as to minimize $n^{1/r} (\ln n)^{2r}$. This implies

$$r = \frac{1}{\sqrt{2}} \sqrt{\frac{\ln n}{\ln \ln n}} + \epsilon, \quad |\epsilon| \leq 1/2$$

and

$$n^{1/r} (\ln n)^{2r} = O(\ln n \exp \sqrt{8 \ln n \ln \ln n}).$$

This finally yields

$$E(T(n)] = O\left(\frac{\sqrt{2r}\, e^{-2r}}{\ln\ln n} \exp\sqrt{8 \ln n \ln\ln n}\right)$$
$$= O(\exp\sqrt{8 \ln n \ln\ln n}). \qquad (5)$$

The asymptotic behavior of this bound is quite attractive for excessively large n: n can be factored within $n^{\epsilon(n)}$ steps with $\epsilon(n) \to 0$ for $n \to \infty$. However, for reasonably sized values the exponent $\epsilon(n)$ is not much smaller than 0.5 and the algorithm therefore hardly beats straightforward factoring algorithms. For instance in the range $n \approx e^{200}$ we choose $r = 4$ and (5) yields $E[T(n)] \leq e^{84} = n^{0.42}$.

Can the above analysis of Dixon's algorithm still be refined leading to a constant in the exponent which is smaller than $\sqrt{8}$? We discuss two main points, (a) the tightness of our lower bound on $\#M(n,v)$ in Lemma 1, (b) the use of more sophisticated factoring algorithms for factoring w over the prime base P in stage 2.

We clearly have $\#M(n,v) \leq \psi(n,v) := \#\{w \in [1,n]$: all prime factors of w are $\leq v\}$. The asymptotic behavior of $\psi(n,v)$ has been analyzed for a long time. De Bruijn (1966) proved

$$\ln\psi(x,y) = \left[\ln\left(1+\frac{y}{\ln x}\right)\frac{\ln x}{\ln y} + \ln\left(1+\frac{\ln x}{y}\right)\frac{y}{\ln y}\right]$$
$$[1 + O(\ln y)^{-1} + O(\ln x)^{-1} + O(1+u)^{-1}]$$

with $u = \ln x / \ln y$. If this upper bound on $\psi(n,v)$ is used instead of the lower bound $\pi(v)^{2r}/(2r)!$ with $v^{2r} \leq n$, it leads to the same constant $\sqrt{8}$ in our time bound. This shows that asymptotically we do not lose too much by the slackness of Lemma 1. However for reasonably sized n the algorithm will perform somewhat better than our rigorous time bound indicates.

Instead of using within stage 2 the straightforward factoring algorithm that leads to Fact 5 we could use one of Pollard's algorithms that finds factors $\leq v$ of n in about $O(\sqrt{v})$ steps. By computational experience, Pollard's ρ-method (1975) detects factors $\leq v$ of n in $O(\sqrt{v}\ln v)$ arithmetical steps mod n, see Guy (1975) and Knuth (1980). This method is highly practical although no rigorous

theoretical time bound is known so far. Recently Brent succeeded in factoring $F_8 = 2^{2^8} + 1$ by a variant of this method. Pollard (1974) also proposed a second method with a rigorous time bound. He computes for sufficiently many small $a \in Z_n^*$, $\gcd\left(\prod_{\nu \le \sqrt{v}} \left(a^{\nu\sqrt{v}} - a^{-\mu}\right), n\right)$ for $\mu = 1, 2, \ldots, \sqrt{v}$. For fixed a these gcd-values can be computed by the fast Fourier transform within $O(\sqrt{v}(\ln v)^2)$ steps. In total, Pollard obtains a worst case time bound $O(v^{0.5+\epsilon})$ for arbitrarily small $\epsilon > 0$, but the constant factor, expressed by O, increases in an unknown way as ϵ decreases. We give a similar but slightly stronger result.

Lemma 2. *For any fixed v the smallest factor $\le v$ of n can be found in $O(\sqrt{v}(\ln v)^2)$ arithmetical steps* $\bmod n$.

Proof. Without loss of generality we assume that $\sqrt{v}$ is an integer. Evaluate $f(x) = \prod_{i=1}^{\sqrt{v}}(x - i) \bmod n$ at $x = t\sqrt{v}$ for $t = 1, 2, \ldots, \sqrt{v}$. Using the fast Fourier-transform this can be done within $O(\sqrt{v}(\ln v)^2)$ arithmetical steps mod n, see e.g. Borodin, Munro (1974), Cor. 4.5.4. Then compute

$$t := \min\{\bar{t} \le \sqrt{v} : \gcd(f(\bar{t}\sqrt{v}), n) > 1\}$$
$$i := \max\{\bar{i} \le \sqrt{v} : (t\sqrt{v} - \bar{i}) \mid n\}$$

Then $t\sqrt{v} - i$ is the smallest factor $\le v$ of n. The correctness of this procedure is obvious. □

Using the above procedure in searching for a prime factor $\le v$ of w in stage 2, we improve Fact 5 to

Fact 6. $T_1(n) = O(\sqrt{v}(\ln v)^2)$.

Now from Facts 1–4, 5, Lemma 1, $v/\ln v \le \pi(v) \le 2v/\ln v$ and Stirling's formula, we obtain for $v = n^{1/2r}$:

$$E[T(n)] = O\left(\frac{\pi(v)\sqrt{v}(\ln v)^2 n(2r)!}{\pi(v)^{2r}} + \pi(v)^3\right)$$
$$= O\left(n^{3/4r} \ln v \sqrt{2r}\, e^{-2r}(\ln n)^{2r} + n^{3/2r}\left(\frac{2r}{\ln n}\right)^3\right). \qquad (6)$$

We choose $r \in \mathcal{N}$ as to minimize $n^{3/4r}(\ln n)^{2r}$ and obtain:

$$2r = \sqrt{\frac{3 \ln n}{2 \ln \ln n}} + \epsilon \quad \text{with} \quad |\epsilon| \le 1$$

$$n^{3/4r}(\ln n)^{2r} \leq \ln n \exp\sqrt{6 \ln n \ln\ln n}$$

$$n^{3/2r}\left(\frac{2r}{\ln n}\right)^3 = O(\exp\sqrt{6 \ln n \ln\ln n}).$$

This finally yields

$$E[T(n)] = O\left(\frac{e^{-2r}(\ln n)^2}{\sqrt{2r}} \exp\sqrt{6 \ln n \ln\ln n}\right)$$
$$= O(\exp\sqrt{6 \ln n \ln\ln n}). \tag{7}$$

Thus we succeeded in decreasing the constant in the exponential term at the expense of increasing the low order factor. In the range $n \approx e^{200}$ we have $r = 6$ and (7) yields $E[T(n)] \approx e^{80}/15 \approx n^{0.4}/15$ which is only marginally better than the conclusion from (5).

Theorem 1. *For each composite n let $E[T(n)]$ be the expected time that the above algorithm finds a proper factor of n. Then for all n*

(1) $E[T(n)] = O(\exp\sqrt{6 \ln n \ln\ln n})$.

(2) The event that the algorithm does not find a proper factor of n within $kE[T(n)]$ steps has probability $\leq 2^{-k}$.

Statement (2) is an immediate consequence of the fact that the distinct events of "test 3" (test 1, resp.) failing" are mutually independent.

A more practical factoring algorithm is obtained if the quadratic residues w in stage 2 are produced via the continuous fraction method (see Morrison and Brillhart, 1975) which implies $w = O(\sqrt{n})$ and if Pollard's ρ-method is used for detecting small prime factors of w.

Under the assumption

(A0) the continuous fraction of $\sqrt{n}$ generates quadratic residues $\bmod n$ which are uniformly distributed in $[1, O(\sqrt{n})]$

the time bound (6) transforms into a time bound

$$E[T(n)] = O\left(n^{3/4r} \ln n\, e^{-r}(\ln n)^r + n^{3/2r}\left(\frac{2r}{\ln n}\right)^3\right) \tag{8}$$

with r even, for the Morrison-Brillhart method. By choosing

$$r = 2\left\lceil \frac{1}{4}\sqrt{\frac{3\ln n}{\ln\ln n}} \right\rceil$$

we obtain

$$n^{3/4r}(\ln n)^r = O((\ln n)^2 \exp\sqrt{3\ln n \ln\ln n})$$

$$n^{3/2r} = O(\exp\sqrt{3\ln n \ln\ln n}).$$

By (8) this implies

Corollary 1. *[Assume (A0).] The Morrison-Brillhart method runs in average time* $O(\exp\sqrt{3\ln n \ln\ln n})$.

In particular (8) with $r = 6$ implies $E[T(n)] \approx e^{56} \approx n^{0.28}$ for $n \approx e^{200}$. However, by experience a well tuned version of the Morrison-Brillhart method behaves somewhat better for reasonably sized n. Wunderlich (1979) obtained average run times $322 \cdot n^{0.152} \approx n^{0.21}$ for $n \approx 10^{40}$. In fact there are several points where our worst case analysis is too pessimistic. The lower bound on $\#M(n,v)$ in Lemma 1 is somewhat too small. Moreover, it is known that the quadratic residues generated by the continuous fraction method can only have prime factors p with $(\frac{n}{p}) = 1$. Since only about half of the primes appear as factors of the w's, this has the effect of doubling the size $\pi(v)$ of the prime base. We estimate that this increases the ratio of w's which are completely factorizable over the prime base by 2^{2r} and therefore causes a speed up factor of about $2^{12} \approx n^{0.041}$ for $n \approx e^{200}$.

Assuming (A0) is only a first imperfect step towards an analysis of the Morrison-Brillhart method. Indeed the continuous fraction of $\sqrt{n}$ behaves too uncivilized. It should be important for a more rigorous analysis to have a lower bound $\#\{p \le v\colon p \text{ prime}, (\frac{n}{p}) = 1\} \ge \frac{v}{c\ln v}$ with $c > 0$ fixed. This would ensure a sufficiently large base of small primes for this method. It is also unclear whether this method finds each factor of n equally likely or whether some factors are harder to find than others. A similar situation will occur in the discussion of an analogous algorithm in Chapter 4.

3. An Analysis and Revision of J. P. C. Miller's Factoring Method.

J. C. P. Miller (1975) proposed a factoring method based on the computation of indices. We shall develop a slightly improved version of Miller's method which turns out to be quite similar to the previously analyzed Dixon algorithm. Under reasonable heuristic assumptions the runtime of our version of Miller's algorithm will be $O(\exp\sqrt{4.5 \ln n \ln\ln n})$. In particular Miller's method does not yield an independent factoring algorithm but merely a specific modification of the method of "combining congruences $\bmod n$". However, as we shall point out, this modification has some decisive advantages in the case that one likes to factor many numbers in the same range. So far all known factoring algorithms collect data which are only useful for factoring one specific number. For instance the congruences collected in Dixon's algorithm cannot be used for different n's. This observation also applies to the factoring algorithms of Morrison–Brillhart (1975), Schroeppel (unpublished, see Monier 1980), Shanks (1971, 1974), and Pollard (1974, 1975). In our version of Miller's method we will collect products of small prime numbers which are near to the number n to be factored. These products of small primes can be uniformly used for factoring all numbers near to n.

For $a \in Z_n^*$, $\operatorname{ord}(a,n) := \min\{\nu \mid a^\nu = 1 \bmod n\}$ is the *order* of $a \bmod n$. $\lambda(n) := \max\{\operatorname{ord}(a,n) \mid a \in Z_n^*\}$ is the order of Z_n^*. Let $h_1, h_2, \ldots, h_\ell$ be a system of independent generators of Z_n^*, then for every $a \in Z_n^*$ there is a representation

$$a = \prod_{i=1}^{\ell} h_i^{m_i} \bmod n$$

where $m_i \bmod \operatorname{ord}(h_i,n)$ is uniquely determined. Then $\operatorname{ind}(a) := (m_1, \ldots, m_\ell)$ is called a (multi-) index of a.

Miller first tries to determine $\operatorname{ord}(a,n)$ for some small primes a as follows. Every solution x of

$$x \cdot \operatorname{ind}(a) = 0 (\bmod \lambda(n)) \tag{1}$$

is a multiple of $\operatorname{ord}(a,n)$. Linear index equations mod $\lambda(n)$ are obtained from representations of n as a sum or a difference of products of small primes. These equations are solved by Gaussian elimination in order to obtain a solution x of (1). We have to factor x in order to determine $\operatorname{ord}(a,n)$. Let $\operatorname{ord}(a,n) = \prod_j a_j^{e_j}$

with a_j prime, then eventually $\gcd(a^{\mathrm{ord}(a,n)/a_j} - 1, n)$ will be a proper factor of n.

As an example, let $n = 1037$.

stage 1: Search for many distinct representations of n or multiples of n as a sum or difference of two products of small primes. For instance we have

$$
\begin{aligned}
* \quad 1037 &= 2^8 5 - 3^5 && \text{i.e.} \quad 2^8 5 = 3^5 \bmod n \\
&= 2^4 \cdot 5 \cdot 13 - 3 && 2^4 \cdot 5 \cdot 13 = 3 \bmod n \\
* \quad &= 2 \cdot 3 \cdot 5^2 \cdot 7 - 13 && 2 \cdot 3 \cdot 5^2 \cdot 7 = 13 \bmod n \\
&= 2^{10} + 13 && 2^{10} = -13 \bmod n \\
* \quad &= 2^2 3^5 + 5 \cdot 13 && 2^2 3^5 = -5 \cdot 13 \bmod n \\
* \quad &= 3 \cdot 7^3 + 2^3 && 3 \cdot 7^3 = -2^3 \bmod n
\end{aligned}
$$

It follows that there exist multi-indices z, a, b, c, d, e for -1, 2, 3, 5, 7, 13 such that

$$
\begin{aligned}
8a + c &= 5b \bmod \lambda(n) \\
4a + c + e &= b \bmod \lambda(n) \\
a + b + 2c + d &= e \bmod \lambda(n) \\
10a &= z + e \bmod \lambda(n) \\
2a + 5b &= z + c + e \bmod \lambda(n) \\
b + 3d &= z + 3a \bmod \lambda(n)
\end{aligned}
$$

stage 2: Gaussian elimination yields

$$120a = 0 \bmod \lambda(n).$$

Hence

$$2^{120} = 1 \bmod n,$$

which means $\mathrm{ord}(2, n) \mid 120$.

The prime factors of 120 are 2, 3, 5 and since $2^{60}, 2^{40}, 2^{24} \neq 1 \bmod n$ we know $\mathrm{ord}(2, n) = 120$.

stage 3: proper factors of n are found as

$$\gcd(2^{60} - 1, n) = 61$$

$$\gcd(2^{40} - 1, n) = 61$$

$$\gcd(2^{24} - 1, n) = 17.$$

The main critical points of this algorithm are the following:

stage 1 How can we generate sufficiently many congruences such that elimination works in stage 2?

stage 2 Suppose a multiple x of $\mathrm{ord}(a, n)$ has been found, what is the chance to find sufficiently many prime factors of x?

stage 3 *will fail to find a proper factor of* $n = \prod_{i=1}^{d} p_i^{l_i}$ *if* $\mathrm{ord}(a, p_i^{l_i})$, $i = 1, \ldots, d$ all coincide.

The following modification circumvents the traps of stages 2 and 3.

In our example for $n = 1037$ we obtain by multiplying the marked congruences:

$$2^{11}3^7 5^3 7^4 = 2^3 3^5 5 \cdot 13^2 \bmod n.$$

Since no prime of our base divides n, this yields

$$2^8 3^2 5^2 7^4 = 13^2 \bmod n.$$

From $2^4 \cdot 3 \cdot 5 \cdot 7^2 = 353 \bmod n$ we obtain

$$353^2 = 13^2 \bmod n$$

which gives us the proper factors

$$\gcd(353 - 13, n) = 17$$

$$\gcd(353 + 13, n) = 61.$$

A formal description of our method is as follows.

begin input n
$v := n^{1/2r}$, $u := n^{d/2r}$
comment the optimal choice of r and d will be made below
Form the list $P = \{p_0, p_1, \ldots, p_{\pi(v)}\}$ of all primes $\leq v$, including $p_0 = -1$

if $\exists p_i \in P,\ i \geq 1$: $p_i \mid n$ then print p_i stop

stage 1 Compute the lists

$$L := \left\{ (w, \underline{a}) \;\middle|\; \begin{array}{l} |w| \leq u, \quad w = \prod_i p_i^{a_i} \\ \underline{a} = (a_i \mid 0 \leq i \leq \pi(v)) \end{array} \right\}$$

$$\tilde{L} := \left\{ (n+w, \underline{b}) \;\middle|\; \begin{array}{l} |w| \leq u, \quad w = \prod_i p_i^{b_i} \\ \underline{b} = (b_i \mid 0 \leq i \leq \pi(v)) \end{array} \right\}$$

$$B := \left\{ (\underline{a}, \underline{b}) \;\middle|\; \begin{array}{l} (\underline{a}, \underline{b}) \not\equiv 0 \bmod 2 \\ \exists w : (w, \underline{a}) \in L \wedge (n+w, \underline{b}) \in \tilde{L} \end{array} \right\}$$

stage 2 Find a nontrivial solution $(f_{(\underline{a},\underline{b})} \mid (\underline{a}, \underline{b}) \in B)$ of

$$\sum_{(\underline{a},\underline{b}) \in B} f_{(\underline{a},\underline{b})} (\underline{a}, \underline{b}) = 0 \bmod 2, \quad f_{(\underline{a},\underline{b})} \in \{0, 1\}. \tag{2}$$

test 2 if no solution exists then increase u goto stage 1

$$x := \prod_{i \leq \pi(v)} p_i^{(\sum_{(\underline{a},\underline{b}) \in B} f_{(\underline{a},\underline{b})} a_i)/2}$$

$$y := \prod_{i \leq \pi(v)} p_i^{(\sum_{(\underline{a},\underline{b}) \in B} f_{(\underline{a},\underline{b})} b_i)/2}$$

comment the construction implies $x^2 = y^2 \bmod n$.

test 3 if $x \not\equiv \pm y \bmod n$ then print $\gcd(x \pm y, n)$ stop
Choose the first $(\underline{a}, \underline{b}) \in B$ such that $f_{(\underline{a},\underline{b})} = 1$
$B := B - \{(\underline{a}, \underline{b})\}$ goto stage 2.

end

This algorithm is virtually very similar to the one of Dixon, and on the other hand it is an improved version of Miller's method. Clearly the linear system (2) has a nontrivial solution as soon as $\#B > 2(\pi(v)+1)$. Compare this with the use of the congruences in Miller's method: if the vectors in B are linearly independent, then Gaussian elimination in Miller's method works as soon as $\#B > \pi(v) + 1$.

However, linearly dependent vectors in B are useless in Miller's method and must be discarded. It is not easy to analyze the ratio of linear dependencies occuring in B. These linear dependencies will speed up our algorithm while they slow down Miller's method.

Even if Gaussian elimination succeeds in Miller's method there are still further traps in stages 2 and 3 of this method, in particular the required factorization of x and $\mathrm{ord}(a, n)$ is a serious obstacle. On the other hand the only remaining trap in our algorithm after solving the linear system is the test "$x \not\equiv \pm y \bmod n$?" Here the argument of Fact 1, Section 2 indicates that this test fails at a frequency 2^{1-d} when n has d distinct prime factors. However we are no more able to provide a rigorous proof.

The time analysis of our algorithm will be based on the following assumptions.

(A1) The ratio of the number of times of "test 3 failing" to "test 3 succeeding" is bounded.

(A2) The numbers which are completely factorizable over P are independently distributed in $[-u, u]$ and $[n-u, n+u]$. These numbers have about the same frequency in $[n-u, n+u]$ and $[0, n]$ for $0 << u << n$.

In particular (A2) implies

$$\#B \geq \psi(n^{d/2r}, n^{1/2r}) \cdot \psi(n, n^{1/2r})/n \geq n^{d/2r}(\ln n)^{-2r-d}.$$

Observe that

$$\begin{aligned}\psi(n, n^{1/r}) &:= \#\{w \in [1, n]\text{: all prime factors of } w \leq v\} \\ &\geq \binom{\pi(n^{1/r})+r}{r} \geq n(\ln n)^{-r} + 2.\end{aligned}$$

Let $T(n)$ be the time of our algorithm. Then (A1), (A2) imply

Fact 7. *$T(n) = O(n^{d/2r} \ln n + n^{3/2r})$ provided $n^{d/2r}(\ln n)^{-2r-d} \geq 2n^{1/2r}$.*

Proof. According to (A1) and (A2), the relation $n^{d/2r}(\ln n)^{-2r-d} \geq 2n^{1/2r}$ implies $\#B > 2(\pi(v)+1)$ and therefore implies the solvability of the linear system (2).

$O(n^{d/2r} \ln n)$ bounds the steps to generate L, $\tilde{L}$, and B, if we compute L (and similarly $\tilde{L}$) as follows. The prime factors $\leq n^{1/2r}$ of w are collected in L_w.

for all w with $|n - w| \leq n^{d/2r}$ do $L_w := \emptyset$
for all $p \in P$ and all ν with $|\nu| \leq n^{d/2r}/p$ do
    [insert p into $L_{n+\nu p - n \bmod p}$]
for every w and every $p_i \in L_w$ do
    [$a_i(w) := \max\{\nu\colon p_i^\nu \mid w\}$]
$L := \{(w, (a_i(w)\colon i \leq \pi(v)) \mid w = \prod_{i \leq \pi(v)} p_i^{a_i(w)}\}$

$O(n^{3/2r})$ bounds the number of steps to solve the linear system (2). ∎

In order to minimize our time bound we choose d, r such that $n^{d/2r} \approx 2(\ln n)^{2r+d} n^{1/2r}$. This yields

$$2r \approx \sqrt{d-1}\sqrt{\frac{\ln n}{\ln\ln n}} \qquad \text{provided } d << r.$$

This yields for $d = 3$:

$$T(n) = O(\exp\sqrt{4.5 \ln n \ln\ln n}).$$

This means that our algorithm is asymptotically superior to Dixon's algorithm, but inferior to the Brillhart–Morrison method. So far we have proved:

Theorem 2. *[Assume (A1), (A2).] The above algorithm has time bound*

$$O(\exp\sqrt{4.5 \ln n \ln\ln n}).$$

One interesting feature of the above algorithm is that the main work in stage 1, namely the construction of the lists L, $\tilde{L}$ is almost independent from n. These lists can be used uniformly for the factorization of all numbers in $[n-u, n+u]$, $u = n^{d/2r}$. In particular, if someone has factored n he already has collected the data to easily factor each number near to n. Considering the problem of factorizing many numbers in $[n-u, n+u]$ we will assume that the lists L, $\tilde{L}$ are built up once for ever and that they are sorted with respect to the first component of the elements $(w, \underline{a})$ and $(n+w, \underline{b})$. Under this assumption we will now bound the remaining number of steps.

Given L and $\tilde{L}$ we can form a sufficiently large subset $\bar{B}$ of B as follows:

$\bar{B} := \emptyset$
while $\#\bar{B} \leq 2(\pi(v) + 1)$ **do**
begin choose $(n + w, \underline{b}) \in \tilde{L}$ at random
eliminate $(n + w, \underline{b})$ from $\tilde{L}$
if $(w, \underline{a}) \in L$ for some $\underline{a}$ **then**
[insert $(\underline{a}, \underline{b})$ into $\bar{B}$]
end

It follows from (A2) that this will take

$$O(\pi(v)(\ln n)^d) = O(n^{1/2r}(\ln n)^d)$$

steps. This yields a total time bound as

$$T(n) = O(n^{1/2r}(\ln n)^d + n^{3/2r})$$

for all r, d with $n^{d/2r}(\ln n)^{-2r-d} \geq 2n^{1/2r}$. We choose r, d such that

$$n^{d/2r}(\ln n)^{-2r-d} \approx 2n^{1/2r}$$

which yields

$$2r \approx \sqrt{d-1}\sqrt{\frac{\ln n}{\ln\ln n}} \qquad \text{provided } d << r.$$

Then minimizing the time bound with respect to d yields

$$d = 2^{2/3}\left(\frac{\ln n}{\ln\ln n}\right)^{1/3}$$

and the corresponding time bound is

$$T(n) = O(\exp(2(\ln n)^{1/3}(\ln\ln n)^{2/3}).$$

Thus we have proved:

Theorem 3. *[Assume (A1), (A2).] Given L, $\tilde{L}$, the time bound of the algorithm is*

$$T(n) = O(\exp(2(\ln n)^{1/3}(\ln\ln n)^{2/3})).$$

This theorem can be interpreted as follows. Suppose we like to factor all numbers in $[n-u, n+u]$, $u = n^{d/2r}$ and let the cost to preprocess the lists L, $\tilde{L}$ be unformly distributed to the numbers in $[n-u, n+u]$. Then the factorization of every specific number in $[n-u, n+u]$ accounts for $O(\exp[2(\ln n)^{1/3}(\ln \ln n)^{2/3}])$ steps.

We observe that the improvement by preprocessing the lists L and $\tilde{L}$ can even be strengthened, if we also preprocess for various k's the lists of all numbers in $[kn-u, kn+u]$ which are completely factorizable over P.

4. Improvements on a Method of Shanks.

Shanks (1971) proposed a factoring method which starts by computing the group of equivalence classes of primitive quadratic forms with discriminant $-n$ and in particular he computes the order $h(-n)$ of this group. Then he factors n by constructing a non-trivial ambiguous class. Under the implicit assumption that the entire group of classes is generated by small "prime" forms, and by neglecting $\log n$ factors, Shanks proves a time bound of about $O(n^{1/4})$. Monier (1980) claims that this time bound can be improved to $O(n^{1/5})$ under the assumption of the generalized Riemann hypothesis. He claims that under this hypothesis the well known convergence

$$\lim_m \frac{\sqrt{n}}{\pi} \prod_{p \le m} \frac{p}{p - (\frac{n}{p})} = h(-n)$$

has an error term $O(n^{1/2}m^{-1/2})$ which would speed up the evaluation of $h(-n)$.

We propose a way to construct ambiguous classes without evaluating $h(-n)$ at all. We exploit the fact that ambiguous forms can be constructed mainly in the same way as we generate solutions of $x^2 = y^2 \bmod n$, by the method of combining congruences. *Under reasonable assumptions this yields an asymptotical time bound* $O(\exp\sqrt{3 \ln n \ln\ln n})$.

We summarize some basic facts on binary quadratic forms. We find it most convenient to follow the original presentation of Gauss (1801, 1889) which slightly differs from that of Shanks (1971). The form $ax^2+2bxy+cy^2$ with $a, b, c \in Z$ will be described by the triple (a, b, c). Two forms (a, b, c) and $(\bar{a}, \bar{b}, \bar{c})$ are <u>equivalent</u> if there exist linear transformations with integer coefficients and determinant 1 transforming the one form into the other; i.e., $T^{-1}\begin{pmatrix} a & b \\ b & c \end{pmatrix} T = \begin{pmatrix} \bar{a} & \bar{b} \\ \bar{b} & \bar{c} \end{pmatrix}$ for some integer matrices T, T^{-1} with $\det T = 1$. Equivalent forms have the same <u>determinant</u> $D := b^2 - ac$. A form (a, b, c) is (properly) <u>primitive</u> if $\gcd(a, 2b, c) = 1$. According to Gauss, the non-primitive forms can all be derived from primitive ones. Therefore it is most important to understand the structure of the primitive forms.

Henceforth we will restrict all considerations to forms with negative determinants $D = b^2 - ac < 0$. In this case the equivalence classes can be characterized by reduced forms. A form (a, b, c) is <u>reduced</u> if $2|b| \le |a| \le |c|$. There is a *gcd*-like algorithm which, given (a, b, c) computes an equivalent reduced form

within $O(\ln|abc|)$ arithmetical steps:

```
while (a, b, c) is not reduced do
    begin b̄ := −b mod c with |b̄| ≤ c/2
          (a, b, c) := (c, b̄, (b̄² − D)/c)
    end
```

Theorem 4. *[Gauss, Artikel 172.] In every equivalence class H with $D < 0$ there is either exactly one reduced form (a, b, c) or exactly two reduced forms $(a, \pm b, c)$. In the latter case, H is called ambiguous.*

A form with $D < 0$ either satisfies $a, c > 0$ or $a, c < 0$. It is called positive in the first and negative in the second case. Positive (negative, resp.) forms $ax^2 + 2bxy + cy^2$ only take positive (negative, resp.) values for real x, y (which follows from $ac > b^2$). Since this property is preserved under the equivalence relation, a class must be either positive, containing only positive forms, or it must be negative and contains only negative forms. Moreover there is a one-one correspondence between the positive and the negative forms as $(a, b, c) \sim (-a, b, -c)$. Therefore we can w.l.o.g. restrict our considerations to positive forms and these generate exactly half of the equivalence classes. The number of equivalence classes with determinant D is finite since a reduced, positive form (a, b, c) always satisfies $2|b| \leq a \leq \sqrt{4|D|/3}$.

Gauss (1801) introduced the composition of (binary) quadratic forms and proved that the equivalence classes with fixed determinant D form an abelian group, say $QF(D)$, under composition. Given two classes H_1, H_2 represented by their reduced forms, the reduced form of $H_1 \cdot H_2$ can be computed within $O(\ln|D|)$ arithmetical steps over numbers $\leq |D|$. The forms which are primitive and positive generate a subgroup of $QF(D)$ which we call $QFP(D)$. The unit element I of the group is represented by $(1, 0, -D)$.

The following assertions are equivalent: (1) H is ambiguous, (2) $H \cdot H = I$, (3) every form (a, b, c) in H is equivalent to $(a, -b, c)$, (4) $T\begin{pmatrix} a & b \\ b & c \end{pmatrix}T^{-1} = \begin{pmatrix} a & b \\ b & c \end{pmatrix}$ for some integer matrices T, T^{-1} with $\det T = -1$.

The reduced form of an ambiguous class is of either of the following three

types:

$$b = 0 \quad \text{or} \quad a = 2b \quad \text{or} \quad a = c.$$

We call these forms ambiguous, they always represent ambiguous classes. These three types of ambiguous forms yield the folowing factorizations of the determinant:

$$-D = ac \quad , \quad -D = b(2c - b) \quad , \quad -D = (a - b)(a + b).$$

In this way the problem of factoring n reduces to the construction of ambiguous forms with determinant $-n$. It is important that Gauss has established a strong correspondence between the factorizations of n and the ambiguous classes in $QFP(-n)$.

We only report the case n odd, since we like to factor only odd nubers.

A pair $(n_1, n_2) \in \mathcal{N}^2$ is an admissible factor pair for n if $n = n_1 \cdot n_2$, $n_1 < n_2$ and $\gcd(n_1, n_2) = 1$. Suppose n has (exactly) ℓ distinct prime factors, then there are (exactly) $2^{\ell-1}$ admissible factor pairs for n.

Theorem 5. *[Gauss, Artikel 257, 258.] Suppose $n \in \mathcal{N}$ is odd and has $\ell \geq 1$ distinct prime factors. Then there are $2^{\ell-1}$ or 2^{ℓ} ambiguous classes in $QFP(-n)$ according to whether $n = 3 \bmod 4$ or $n = 1 \bmod 4$. Each of the $2^{\ell-1}$ admissible factor pairs of n is obtained by the reduced form of exactly one in case $n = 3 \bmod 4$ (two in case $n = 1 \bmod 4$) of these ambiguous classes.*

Example. We list n: all ambiguous forms with determinant $-n$ and $b \geq 0$ that are primitive, reduced, and positive; the corresponding list of admissible factor pairs.

$n = 3$: $(1,0,3)$; $(1,3)$
$n = 5$: $(1,0,5)$, $(2,1,3)$; $(1,5)$, $(1,5)$
$n = 15$: $(1,0,15)$, $(3,0,5)$; $(1,15)$, $(3,5)$
$n = 21$: $(1,0,21)$, $(3,0,7)$, $(2,1,11)$, $(5,2,5)$; $(1,21)$, $(3,7)$, $(1,21)$, $(3,7)$
$n = 105$: $(1,0,105)$, $(3,0,35)$, $(5,0,21)$, $(7,0,15)$, $(2,1,53)$, $(6,3,19)$, $(10,5,13)$, $(11,4,11)$; $(1,105)$, $(3,35)$, $(5,21)$, $(7,15)$, $(1,105)$, $(3,35)$, $(5,21)$, $(7,15)$.

The distinction between the cases $n = 1 \bmod 4$ and $n = 3 \bmod 4$ is explained as follows. The ambiguous and reduced form $(2, 1, (n+1)/2)$ is primitive in case

$n = 1 \bmod 4$ whereas it is imprimitive in case $n = 3 \bmod 4$, since in the latter case $\gcd(2, 2, (n+1)/2) = 2$. Since the product of two ambiguous classes is again ambiguous, there must be twice as many ambiguous classes in case $n = 1 \bmod 4$ as there are in case $n = 3 \bmod 4$.

The remaining point to be discussed for the factorization of n is how to generate ambiguous classes in $QFP(-n)$. This will be done by exploiting the group structure of $QFP(-n)$. Let $H, \bar{H} \in QFP(-n)$ be represented by (a, b, c) and $(\bar{a}, \bar{b}, \bar{c})$, i.e., $H = [(a, b, c)]$, $\bar{H} = [(\bar{a}, \bar{b}, \bar{c})]$. Then by definition a representative (A, B, C) for $H \cdot \bar{H}$ can be found as follows:

$\mu := \gcd(a, \bar{a}, b + \bar{b})$
Compute $\alpha, \beta, \gamma \in Z$ such that
$\alpha a + \beta \bar{a} + \gamma(b + \bar{b}) = \mu$.
$A := a\bar{a}/\mu^2$
$B := [\alpha a\bar{b} + \beta \bar{a} b + \gamma(b\bar{b} - n)]/\mu \bmod A$
$C := (n + B^2)/A$

In the special case that $\gcd(a, \bar{a}) = 1$ one obtains in this way (observe that we can choose $\gamma = 0$ and α, β such that $\alpha a + \beta \bar{a} = 1$):

$A := a\bar{a}$
Choose B such that
$B = b \bmod a$ and $B = \bar{b} \bmod \bar{a}$
$C := (n + B^2)/A$.

(A, B, C) will be primitive but not necessarily reduced. $[(A, B, C)]$ does not depend on the distinct possible choices for α, β, γ, B, and C. Since α, β, γ can be computed via Euclid's gcd-algorithm, it is clear that this multiplication scheme requires only $O(\ln n)$ arithmetical steps over numbers $\leq O(n)$ provided (a, b, c) and $(\bar{a}, \bar{b}, \bar{c})$ are reduced. It can easily be seen that

$$[(a, b, c)][(a, -b, c)] = I.$$

In this case $\mu = a$, $A = 1$ and the choice $\alpha = 1$, $\beta = \gamma = 0$ yields $B = ab$ and therefore $A \mid B$. Then $A = 1$ implies $[(A, B, C)] = I$.

The special case $\gcd(a, \bar{a}) = 1$ of this multiplication scheme immediately implies the following.

Fact 8. *Let $[(a,b,c)] \in QFP(-n)$ and let $a = \prod_i p_i^{\alpha_i}$ be the prime factorization of a, then $[(a,b,c)] = \prod_i [(p_i^{\alpha_i}, b_i, c_i)]$ with $b_i := b \bmod p_i^{\alpha_i}$ and $c_i := (b_i^2+n)/p_i^{\alpha_i}$.*

The possibly occuring factors $[(p_i^{\alpha_i}, b_i, c_i)]$ in Fact 8 can be characterized as follows.

Lemma 3. *Let p be prime, $p \neq 2$, $\gcd(p,n) = 1$ and $\alpha \geq 1$. There exists $[(p^\alpha, b, c)] \in QFP(-n)$ with integers b, c iff $(\frac{-n}{p}) = 1$. If $(\frac{-n}{p}) = 1$ there are exactly two of these classes, namely $[(p^\alpha, \pm b, (n+b^2)/p^\alpha)]$ for b with $b^2 = -n \bmod p^\alpha$.*

Proof. Suppose (p^α, b, c) is a positive form with determinant $-n$. Then $-n = b^2 - p^\alpha c$ which means that $-n$ is a quadratic residue mod p^α. Hence $(\frac{-n}{p^\alpha}) = (\frac{-n}{p}) = 1$. There are exactly two square roots $\pm b$ of $-n \bmod p^\alpha$. The classes $[(p^\alpha, \pm b, (n+b^2)/p^\alpha)]$ are distinct and primitive. In fact these classes are inverse and non-ambiguous, since $\gcd(p,n) = 1$, $p \neq 2$. ∎

We denote one of the classes $[(p^\alpha, \pm b, (n+b^2)/p^\alpha)]$ occuring in Lemma 3 as $I_{p^\alpha,n}$. Then the other class must be $(I_{p^\alpha,n})^{-1}$. It is clear from the multiplication scheme that

$$\{(I_{p,n})^\alpha, (I_{p,n})^{-\alpha}\} = \{I_{p^\alpha,n}, (I_{p^\alpha,n})^{-1}\}.$$

This implies that Fact 8 can be rewritten as follows.

Lemma 4. *Let $[(a,b,c)] \in QFP(-n)$, a odd and let $a = \prod p_i^{\alpha_i}$ be the prime factorization of a. Then*

$$[(a,b,c)] = \prod_i (I_{p_i,n})^{\alpha_i \cdot \epsilon_i} \qquad \text{with } \epsilon_i = \pm 1.$$

In particular, factoring $[(a,b,c)] \in QFP(-n)$ as in Lemma 4 can be done roughly in the time which is necessary to factor a. Since we know

$$(I_{p_i,n})^{\alpha_i \epsilon_i} = [(p_i^{\alpha_i}, b_i, c_i)]$$

with $b_i = b \bmod p_i^{\alpha_i}$, $c_i = (b_i^2 + n)/p_i^{\alpha_i}$, we can easily check whether $\epsilon_i = 1$ or $\epsilon_i = -1$. Also, in the case that a is even, c must be odd provided (a,b,c) is

primitive. Hence, if a is even we can apply Lemma 4 to the form $(c, -b, a)$ which is equivalent to (a, b, c).

By means of Lemma 4 we can generate ambiguous forms with determinant $-n$ mainly in the same way as congruences $x^2 = y^2 \bmod n$ are produced by Dixon's factoring algorithm.

Construction of ambiguous classes in $QFP(-n)$.

stage 1 Construct a factor base

$P := \{p \mid 2 < p \leq v,\ p \text{ prime}, \left(\frac{-n}{p}\right) = 1\}$

if $\exists p \in P$: $p \mid n$ then print p stop

for all $p \in P$ compute $I_p := (p, b, (b^2 + n)/p)$

comment [We discuss the optimal choice of v below. Compute I_p by solving $b^2 = -n \bmod p$ using the probabilistic algorithm of Berlekamp, Rabin, see Rabin (1979).]

stage 2 Choose a random $H \in QFP(-n)$ which is generated by the I_p with $p \in P$ [i.e., compute $H = \prod_{p_i \in P} I_{p_i}^{a_i}$ with a random $(a_i \mid p_i \in P) \in \mathcal{N}^{\#P}$ such that $\sum_i \ln a_i \leq (\ln n)^2$]

Compute the reduced form (a, b, c) of $H \cdot H$.

Try to factor a over P and $[(a, b, c)]$ over $\{I_p \mid p \in P\}$.

if $H^2 = \prod_i I_{p_i}^{a_i}$ with $a_i \in Z$ then

[store $\underline{a} = (a_i \mid p_i \in P)$ and set $H_{\underline{a}} := H$]

while $\leq \#P$ vectors $\underline{a}$ have been found goto stage 2

Solve $\sum_{\underline{a}} f_{\underline{a}} \underline{a} = 0 \bmod 2$ with $f_{\underline{a}} \in \{0, 1\}$

Then $\prod_{f_{\underline{a}}=1} H_{\underline{a}} \prod_{p_i \in P} I_{p_i}^{-(\sum_{\underline{a}} f_{\underline{a}} a_i)/2}$ is an ambiguous class.

comment Observe that the construction implies $\prod_{f_{\underline{a}}=1} H_{\underline{a}}^2 = \prod_{p_i \in P} I_{p_i}^{\sum_{\underline{a}} f_{\underline{a}} a_i}$

Even if $n = 1 \bmod 4$ we do not include $p = 2$ into the factor base P, since the ambiguous class $I_2 = [(2, 1, (n+1)/2)]$ corresponds to the trivial factor pair $(1, n)$.

Example. $n = 1037$

We choose the factor base $P = \{3, 13\}$, we have $(\frac{n}{5}) = (\frac{n}{7}) = (\frac{n}{11}) = -1$. The corresponding classes are

$$I_3 = [(3, 1, 346)] \quad , \quad I_{13} = [(13, 4, 81)].$$

One obtains

$$I_3^4 = I_{13}^{-1}$$

$$I_{13}^2 = I_3^2.$$

Hence $I_{13} \cdot I_3^{-1}$ is ambiguous. The reduced form in this class is $(34, 17, 39)$ which yields the factorization

$$1037 = 17(78 - 17) = 17 \cdot 61.$$

Observe that the factor base in this example is smaller than in the application of Miller's method in Section 3. Dixon's algorithm would require a larger factor base too. Indeed the factor base is so small since the primes $p = 5, 7, 11$ are excluded because $(\frac{n}{p}) = -1$.

In our analysis of the algorithm we will use the following heuristic assumptions.

(A3) $\#\{p \leq v : p \text{ prime}, (\frac{n}{p}) = 1\} \geq \frac{v}{c \ln v}$ with $c > 0$ fixed.

(A4) every admissible factor pair of n corresponds to some ambiguous class which is generated by the I_p, $p \leq v$.

Assumption (A3) certainly fails for a few n but it must hold for most n since we have:

$$\frac{1}{\#Z_p^*} \sum_{a \in Z_p^*} \#\{p \leq v, p \text{ prime}, (\tfrac{a}{p}) = 1\} \approx \frac{v}{2 \ln v}.$$

This follows from $\pi(v) \approx v / \ln v$ and from the fact that $(\frac{a}{p}) = 1$ for exactly half of the $a \in Z_p^*$. We argue that this supports (A3) since we can as well apply our algorithm to factor any number $n \cdot k$, k odd, $k << n$. Then factors of n will be found with the same probability as those of k.

The assumption (A4) is still somewhat weaker than the assumption used by Shanks (1971) that the whole group $GFP(-n)$ is generated by the classes I_p with small p.

Under the assumptions (A3), (A4) the analysis of the algorithm becomes virtually very similar to the analysis of Dixon's algorithm. The main advantage over Dixon's algorithm is that we have to factor numbers $a = O(\sqrt{n})$, instead of numbers $w = O(n)$, over the base of small primes. Therefore we can argue as in the case, that quadratic residues mod n, $w = O(\sqrt{n})$ are constructed by the continuous fraction method, see the send of Section 2. We choose

$$v = n^{1/2r} \quad , \quad r = 2\left\lceil \frac{1}{4}\sqrt{\frac{3 \ln n}{\ln \ln n}} \right\rceil$$

and obtain n as a final result.

Theorem 4. *[Assume (A3), (A4).] Suppose we factor a composite n via the construction of ambiguous forms with determinant $-n$ as above, then for each n a proper factor of n will be found with probability $1/2$ within $O(\exp \sqrt{3 \ln n \ln \ln n})$ steps.*

The above factoring method can be interpreted as the continuous fraction method in case of negative determinants. Conversely, in case of positive determinants $D = b^2 - ac > 0$, there is a different concept of reduced forms and there are many equivalent reduced forms. According to Gauss, Artikel 183–187, the equivalent reduced forms can be developed into an even and symmetric period. The recursion for developing this period is the same as that for evaluating the period of the continuous fraction of $\sqrt{D}$. Shanks exploited this coincidence and proposed an algorithm to factor n by constructing an ambiguous form with positive determinant n. Shanks has a way to make giant steps within the period of equivalent reduced forms (this is used in order to decide whether two forms are equivalent). This second algorithm of Shanks runs in about $O(n^{1/4})$ steps, see Monier (1980) for a more detailed exposition of this method.

Acknowledgement. I am greatly indebted to the Stanford Computer Science Department whose support enabled this work. In particular I thank D. Knuth for many hints and his efficient cooperation. I thank J. Vuillemin for communicating to me the thesis of L. Monier.

References

[1] Brent, R. P., "Analysis of some new cycle finding and factorization algorithms." Department of Computer Science, Australian National University, (1979).

[2] de Bruijn, N. G., "On the number of positive integers $\leq x$ and free of prime factors $> y$, II," *Nederl. Akad. Wetensch. Proc. Ser. A* 69 (1966), 239–247.

[3] Diffie, W. and Hellman, M., "New directions in cryptography," *IEEE Trans. Information Theory* IT-22 (1976), 644–654.

[4] Dixon, J. D., "Asymptotical fast factorization of integers." Report, Department of Mathematics, Carleton University, Ottawa (1978).

[5] Gauss, C. F., *Disquisitiones Arithmeticae*, Leipzig (1801). German translation: *Untersuchungen über höhere Arithmetik*, Springer, Berlin (1889).

[6] Guy, R. K., "How to factor a number," *Proc. Fifth Manitoba Conference on Numerical Math.* (1975), 49–90.

[7] Knuth, D. E., *The Art of Computer Programming*, Volume 2, *Seminumerical Algorithms*, Addison-Wesley (1969), second edition (1980).

[8] Knuth, D. E. and Trabb Pardo, L, "Analysis of a simple factorization algorithm," *Theoretical Computer Science* 3 (1976), 321–348.

[9] Legendre, A. M., *Theorie des Nombres* Tome I, Paris (1798), reprint Blanchard, Paris (1955).

[10] Miller, J. C. P., "On factorization, with a suggested new approach," *Math. Computation* 29 (1975), 155–172.

[11] Monier, L, "Algorithmes de factorisation d'entiers," Thèse d'informatique, Université Paris Sud (1980).

[12] Morrison, M. A. and Brillhart, J., "A method of factorization and the factorization of F_7," *Math. Computation* 29 (1975), 183–205.

[13] Pollard, J. M., "Theorems on factorization and primality testing," *Proc. Cambridge Phil. Soc.* 76 (1976), 521–528.

[14] Pollard, J. M., "A Monte Carlo method for factorization," *BIT* 15 (1975), 331–334.

[15] Rabin, M. O., "Probabilistic algorithms in finite fields," *SIAM J. Comp.* 9, 2 (1980), 273–280.

[16] Rivest, R. L, Shamir, A., and Adleman, L., "A method for obtaining digital signatures and public key cryptosystems," *Comm. ACM* 21, 2 (1978), 120–126.

[17] Rivest, R. L. and Pinter, R. Y., "Using hyperbolic tangents in integer factoring," MIT Report (1979).

[18] Shanks, D., "Class number, a theory of factorization and genera," *Proc. Symp. Pure Math., American Math. Soc.* 20 (1971), 415–440.

[19] Shanks, D., "The infrastructure of real quadratic fields and its application," *Proc. Boulder Number Theory Conference*, University of Colorado (1972), 217–224.

[20] Wunderlich, M. C., "A running time analysis of Brillhart's continuous fraction method," in Springer, *Lecture Notes in Math.* 751 (1979), 328–342.